Study guide: Nonlinear differential equation problems

Hans Petter Langtangen[1]

Center for Biomedical Computing, Simula Research Laboratory and Department of Informatics, University of Oslo[1]

Nov 5, 2014

What makes a differential equations nonlinear?

- In linear differential equations, the unknown u or its derivatives appear in linear terms $au(t)$, $au'(t)$, $a\nabla^2 u$, where a is independent of u.
- All other types of terms containing u are *nonlinear* and contain products of u or its derivatives.

Linear ODE:

$$u'(t) = a(t)u(t) + b(t)$$

Nonlinear ODE:

$$u'(t) = u(t)(1 - u(t)) = u(t) - {\color{red}u(t)^2}$$

This (pendulum) ODE is also nonlinear:

$$u'' + \gamma \sin u = 0$$

1. Introduction of basic concepts

Introduction of basic concepts

- Logistic ODE as simple model for a nonlinear problem
- Introduction of basic techniques:
  - Explicit time integration (no nonlinearities)
  - Implicit time integration (nonlinearities)
  - Linearization and Picard iteration
  - Linearization via Newton's method
  - Linearization via a trick like geometric mean
- Numerical examples

The scaled logistic ODE

$$u'(t) = u(t)(1 - u(t)) = u - u^2$$

Linearization by explicit time discretization

Forward Euler method:

$$\frac{u^{n+1} - u^n}{\Delta t} = u^n(1 - u^n),$$

which is a *linear* algebraic equation for the unknown value u^{n+1}.

Explicit time integration methods will (normally) linearize a nonlinear problem.

Another example: 2nd-order Runge-Kutta method

$$\begin{aligned}
u^* &= u^n + \Delta t u^n(1 - u^n),\\
u^{n+1} &= u^n + \Delta t \frac{1}{2}\left(u^n(1 - u^n) + u^*(1 - u^*)\right) .
\end{aligned}$$

An implicit method: Backward Euler discretization

Use backward time difference:

$$\frac{u^n - u^{n-1}}{\Delta t} = u^n(1 - u^n)$$

This is a nonlinear algebraic equation for the unknown u^n! The equation is of quadratic type (which can easily be solved exactly):

$$\Delta t(u^n)^2 + (1 - \Delta t)u^n - u^{n-1} = 0\,.$$

Detour: new notation

To make formulas less overloaded and the mathematics as close as possible to computer code, a new notation is introduced:

- $u^{(1)}$ is the value of the unknown at the previous time level
- In general: $u^{(\ell)}$ is the value of the unknown ℓ levels back in time
- u denotes the unknown to be solved for
- Backward Euler method: u for u^n, $u^{(1)}$ for u^{n-1}

Nonlinear equation to solve:

$$F(u) = \Delta t u^2 + (1 - \Delta t)u - u^{(1)} = 0$$

Exact solution of nonlinear equations

Solution of $F(u) = 0$:

$$u = \frac{1}{2\Delta t}\left(-1 - \Delta t \pm \sqrt{(1-\Delta t)^2 - 4\Delta t u^{(1)}}\right)$$

Warning

Nonlinear algebraic equations may have multiple solutions!

How do we pick the right solution? Let's investigate the nature of the two roots:

```
>>> import sympy as sp
>>> dt, u_1, u = sp.symbols('dt u_1 u')
>>> r1, r2 = sp.solve(dt*u**2 + (1-dt)*u - u_1, u)  # find roots
>>> r1
(dt - sqrt(dt**2 + 4*dt*u_1 - 2*dt + 1) - 1)/(2*dt)
>>> r2
(dt + sqrt(dt**2 + 4*dt*u_1 - 2*dt + 1) - 1)/(2*dt)
>>> print r1.series(dt, 0, 2)
-1/dt + 1 - u_1 + dt*(u_1**2 - u_1) + O(dt**2)
>>> print r2.series(dt, 0, 2)
u_1 + dt*(-u_1**2 + u_1) + O(dt**2)
```

Linearization

- In general, we cannot solve nonlinear algebraic equations with formulas
- We must *linearize* the equation, or create a recursive set of *linearized* equations whose solutions hopefully converge to the solution of the nonlinear equation
- Manual linearization may be an art
- Automatic linearization is possible (cf. Newton's method)

Examples will illustrate the points!

Picard iteration

Let us write the quadratic nonlinear equation, arising from Backward Euler discretization of the logistic ODE, in a more compact form

$$F(u) = au^2 + bu + c = 0$$

Let u^- be an available approximation of the unknown u. Then we can linearize the term u^2 simply by writing $u^- u$. The resulting equation, $\hat{F}(u) = 0$, is now linear:

$$F(u) \approx \hat{F}(u) = au^- u + bu + c = 0$$

Problem: the solution u of $\hat{F}(u) = 0$ is not the exact solution of $F(u) = 0$.

Idea: Set $u^- = u$ and repeat the procedure.

The idea of turning a nonlinear equation into a linear one by using an approximation u^- of u in nonlinear terms is a widely used

Picard iteration

At a time level, set $u^- = u^{(1)}$ (solution at previous time level) and iterate:

$$u = -\frac{c}{au^- + b}, \quad u^- \leftarrow u\,.$$

This technique is known as

- fixed-point iteration
- successive substitutions
- nonlinear Richardson iteration
- Picard iteration

Using subscripts as in real math books: u^k is computed approximation in iteration k and u^{k+1} is the next approximation:

$$au^k u^{k+1} + bu^{k+1} + c = 0 \quad \Rightarrow \quad u^{k+1} = -\frac{c}{au^k + b}, \quad k = 0, 1, \ldots$$

or

Stopping criteria

Using change in solution:

$$|u - u^-| \leq \epsilon_u,$$

or change in residual:

$$|F(u)| = |au^2 + bu + c| < \epsilon_r \, .$$

A single Picard iteration

Common simple and cheap technique: perform 1 single Picard iteration

$$\frac{u^n - u^{n-1}}{\Delta t} = u^n(1 - u^{n-1})$$

Inconsistent discretization - can produce quite inaccurate results, but is very popular.

Implicit Crank-Nicolson discretization

Crank-Nicolson discretization:

$$[D_t u = u(1-u)]^{n+\frac{1}{2}}$$

Written out:

$$\frac{u^{n+1}-u^n}{\Delta t} = u^{n+\frac{1}{2}} - (u^{n+\frac{1}{2}})^2$$

Approximate $u^{n+\frac{1}{2}}$ as usual by an arithmetic mean,

$$u^{n+\frac{1}{2}} \approx \frac{1}{2}(u^n + u^{n+1}),$$

The same arithmetic mean applied to the nonlinear term gives

$$(u^{n+\frac{1}{2}})^2 \approx \frac{1}{4}(u^n + u^{n+1})^2,$$

Linearization by a geometric mean

Using a *geometric mean* for $(u^{n+\frac{1}{2}})^2$ linearizes the nonlinear term $(u^{n+\frac{1}{2}})^2$ (error $\mathcal{O}(\Delta t^2)$ as in the discretization of u'):

$$(u^{n+\frac{1}{2}})^2 \approx u^n u^{n+1}$$

Arithmetic mean on the linear $u^{n+\frac{1}{2}}$ term and a geometric mean for $u^{n+\frac{1}{2}})^2$ gives a linear equation for u^{n+1}:

$$\frac{u^{n+1} - u^n}{\Delta t} = \frac{1}{2}(u^n + u^{n+1}) + u^n u^{n+1}$$

Note: Here we turned a nonlinear algebraic equation into a linear one. No need for iteration!

Newton's method

Write the nonlinear algebraic equation as

$$F(u) = 0$$

Newton's method: linearize $F(u)$ by two terms from the Taylor series,

$$\begin{aligned} F(u) &= F(u^-) + F'(u^-)(u - u^-) + \frac{1}{2}F''(u^-)(u - u^-)^2 + \cdots \\ &\approx F(u^-) + F'(u^-)(u - u^-) = \hat{F}(u)\,. \end{aligned}$$

The linear equation $\hat{F}(u) = 0$ has the solution

$$u = u^- - \frac{F(u^-)}{F'(u^-)}\,.$$

Or with an iteration index: